**Kubernetes**

**main.tf**

terraform {

required\_version = ">= 1.5.0"

required\_providers {

aws = {

source = "hashicorp/aws"

version = "~> 5.0"

}

}

}

provider "aws" {

region = var.region

}

# EKS cluster with one managed node group (2 nodes)

module "eks" {

source = "terraform-aws-modules/eks/aws"

version = "~> 20.0"

cluster\_name = var.cluster\_name

cluster\_version = "1.29"

vpc\_id = var.vpc\_id

subnet\_ids = var.private\_subnet\_ids

enable\_irsa = true

eks\_managed\_node\_groups = {

default2 = {

min\_size = 2

max\_size = 3

desired\_size = 2

instance\_types = ["t3.medium"]

capacity\_type = "ON\_DEMAND"

}

}

}

# To let kubectl know where the cluster is

data "aws\_eks\_cluster" "this" {

name = module.eks.cluster\_name

depends\_on = [module.eks]

}

data "aws\_eks\_cluster\_auth" "this" {

name = module.eks.cluster\_name

}

output "cluster\_name" { value = module.eks.cluster\_name }

output "cluster\_endpoint" { value = data.aws\_eks\_cluster.this.endpoint }

output "cluster\_certificate" { value = data.aws\_eks\_cluster.this.certificate\_authority[0].data }

output "nodegroup\_role\_arn" { value = module.eks.eks\_managed\_node\_groups["default2"].iam\_role\_arn }

# Access entry for clahaan-user6

resource "aws\_eks\_access\_entry" "clahaan\_user5" {

cluster\_name = module.eks.cluster\_name # directly put your cluster name

principal\_arn = "arn:aws:iam::909688465000:user/clahaan-user5"

type = "STANDARD"

}

# Attach Cluster Admin policy

resource "aws\_eks\_access\_policy\_association" "clahaan\_user5\_cluster\_admin" {

cluster\_name = module.eks.cluster\_name # directly put your cluster name

policy\_arn = "arn:aws:eks::aws:cluster-access-policy/AmazonEKSClusterAdminPolicy"

principal\_arn = aws\_eks\_access\_entry.clahaan\_user5.principal\_arn

access\_scope {

type = "cluster"

}

}

**terraform.tfvars**

region = "ap-south-1"

cluster\_name = "Pra-eks-node"

vpc\_id = "vpc-07ea395ad99b60b99"

private\_subnet\_ids = ["subnet-053773abd414e0b3a", "subnet-06c6453d2125124b4"]

public\_subnet\_ids = ["subnet-0963bfb6f84925488", "subnet-09eeb3e63141439cf"]

**variable.tf**

variable "region" {

type = string

default = "ap-south-1"

}

variable "cluster\_name" {

type = string

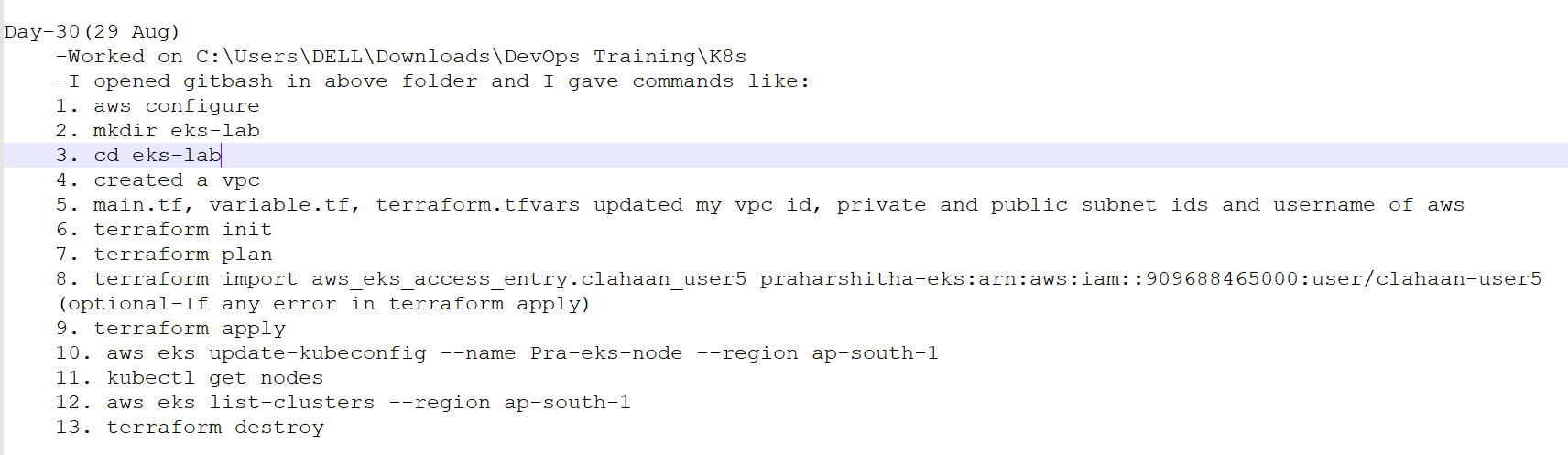
default = "Pra-eks-node"

}

variable "vpc\_id" { type = string }

variable "private\_subnet\_ids" { type = list(string) }

variable "public\_subnet\_ids" { type = list(string) }



1. aws configure

2. mkdir eks-lab

3. cd eks-lab

4. created a vpc

5. main.tf, variable.tf, terraform.tfvars updated my vpc id, private and public subnet ids and username of aws

6. terraform init

7. terraform plan

8. terraform import aws\_eks\_access\_entry.clahaan\_user5 praharshitha-eks:arn:aws:iam::909688465000:user/clahaan-user5

(optional-If any error in terraform apply)

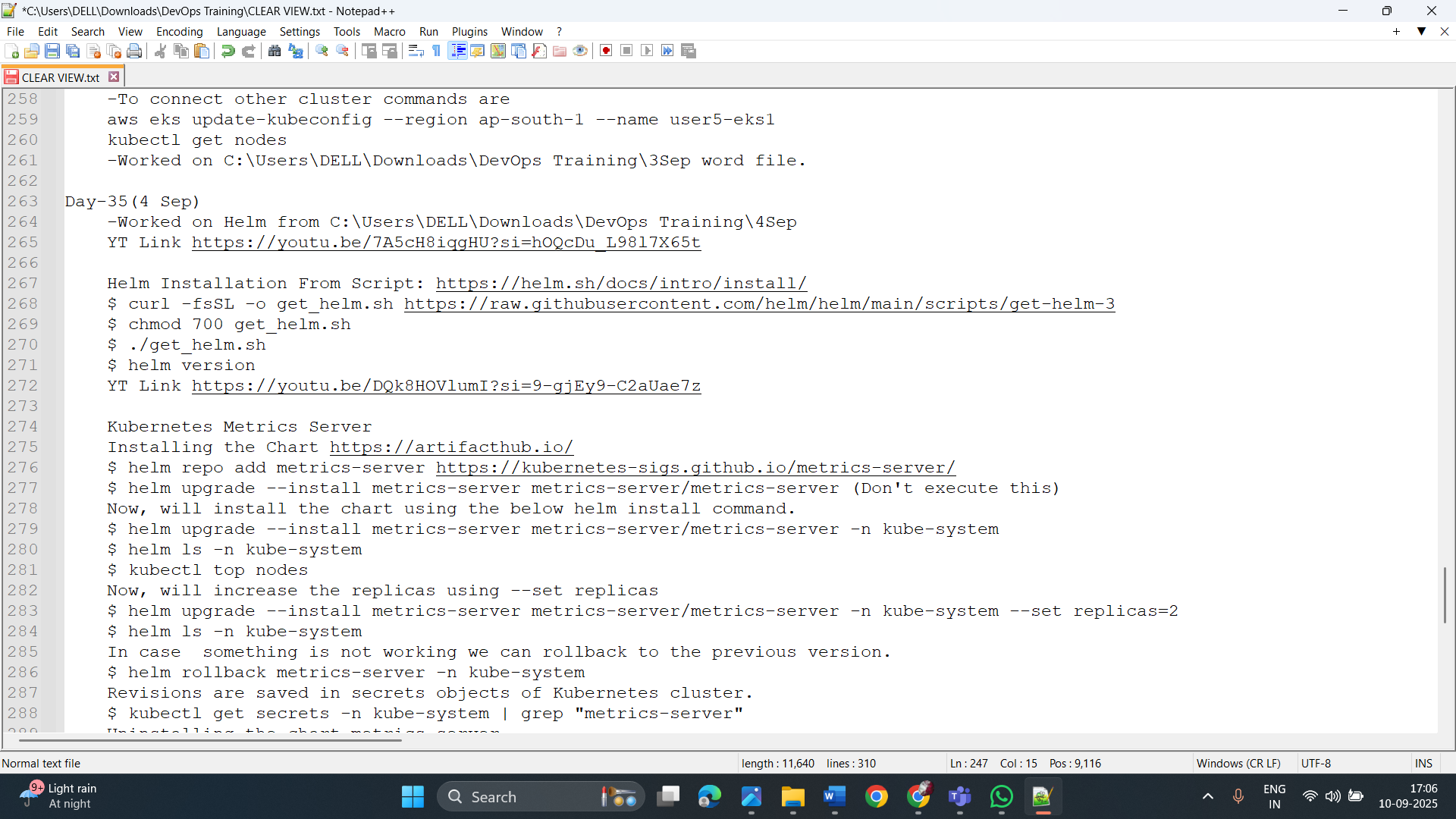
9. terraform apply

10. aws eks update-kubeconfig --name Pra-eks-node --region ap-south-1

11. kubectl get nodes

12. aws eks list-clusters --region ap-south-1

13. terraform destroy



Day-34 (3 Sep)

-To connect other cluster commands are

aws eks update-kubeconfig --region ap-south-1 --name user5-eks1

kubectl get nodes

-Worked on C:\Users\DELL\Downloads\DevOps Training\3Sep word file.

Day-35(4 Sep)

-Worked on Helm from C:\Users\DELL\Downloads\DevOps Training\4Sep

YT Link https://youtu.be/7A5cH8iqgHU?si=hOQcDu\_L98l7X65t

Helm Installation From Script: https://helm.sh/docs/intro/install/

$ curl -fsSL -o get\_helm.sh https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3

$ chmod 700 get\_helm.sh

$ ./get\_helm.sh

$ helm version

YT Link https://youtu.be/DQk8HOVlumI?si=9-gjEy9-C2aUae7z

Kubernetes Metrics Server

Installing the Chart https://artifacthub.io/

$ helm repo add metrics-server https://kubernetes-sigs.github.io/metrics-server/

$ helm upgrade --install metrics-server metrics-server/metrics-server (Don't execute this)

Now, will install the chart using the below helm install command.

$ helm upgrade --install metrics-server metrics-server/metrics-server -n kube-system

$ helm ls -n kube-system

$ kubectl top nodes

Now, will increase the replicas using --set replicas

$ helm upgrade --install metrics-server metrics-server/metrics-server -n kube-system --set replicas=2

$ helm ls -n kube-system

In case something is not working we can rollback to the previous version.

$ helm rollback metrics-server -n kube-system

Revisions are saved in secrets objects of Kubernetes cluster.

$ kubectl get secrets -n kube-system | grep "metrics-server"

Uninstalling the chart metrics-server.

$ helm uninstall metrics-server -n kube-system

-Then creating chart C:\Users\DELL\Downloads\DevOps Training\4Sep from Helm-for-Freshers-Step-by-Step-Guide PDF file.